****

**Digital Library**

**שם התלמיד: יונתן הררי**

**מספר תעודת זהות: 326544269**

**שם המנחה: יורם אביטוב**

**תאריך הגשה: 23.4.2022**

**מרכיבים נדרשים בפרויקט (חובה):**

1. **תקשורת** – על הפרויקט לכלול תקשורת בין רכיבים שונים, בדגש על שרת המרכזי המקבל מידע ממספר לקוחות.
2. **אבטחת מידע** – על הפרויקט לכלול שכבה של אבטחת מידע, בדגש על הצפנת תקשורת והגנה על מידע רגיש בעת כתיבה למסד נתונים ותקשורת באינטרנט
3. **מסד נתונים** – יכול להיות פשוט או מורכב בהתאם לצרכי הפרויקט.
4. **מערכות הפעלה** – על הפרויקט לכלול פעולות ברמת מעה"פ כפי שנלמד בשיעורים, בדגש על עבודה עם Multi-Processes ו Multi-Threads (תכנות מקבילי ואסינכרוני), תקשורת בין תהליכים, וסנכרון משאבים משותפים
5. **חלק ויזואלי (GUI)** – על הפרויקט לכלול שכבת תצוגה המאפשרת שימוש נח וברור למשתמשים השונים.
6. **תיעוד** - עבור כל פונקציה יש לפרט את כניסות, יציאות ומה הפונקציה מבצעת. בנוסף יש לתעד את מבני הנתונים בהם משתמשים. על הקוד לכלול תיעוד כללי המתאר את תהליך קבלת ההחלטות. יש לשים דגש על בדיקת נכונות הקלט ונכונות הבדיקות הלוגיות.

**שם הפרויקט**

**תוכן עניינים מומלץ לספר הפרויקט**

**תוכן עניינים**

[1 ניהול שינויים 4](#_Toc101641769)

[2 מבוא 5](#_Toc101641770)

[2.1 תיאור המוצר המוגמר 5](#_Toc101641771)

[2.2 נושא המחקר בפרויקט 5](#_Toc101641772)

[3 סביבת העבודה בפרויקט 6](#_Toc101641773)

[3.1 טכנולוגיות בשימוש בפרויקט 8](#_Toc101641774)

[3.2 מדריך למשתמש 8](#_Toc101641775)

[4 אפיון דרישות וארכיטקטורת המערכת 9](#_Toc101641776)

[4.1 דרישות ושימושי מערכת – Use Cases 9](#_Toc101641777)

[4.2 סביבת הפרויקט – Eco – System 10](#_Toc101641778)

[4.3 ארכיטקטורת המערכת 11](#_Toc101641779)

[4.4 ממשק משתמש - GUI 12](#_Toc101641780)

[5 מדריך למפתח 13](#_Toc101641781)

[5.1 הרצת השרת 13](#_Toc101641782)

[5.2 דיאגרמת התיקיות בפרוייקט ותפקידיהם 14](#_Toc101641783)

[5.3 דיאגרמת UML של כל מחלקות הפרויקט והתלויות ביניהן 16](#_Toc101641784)

[5.4 רשימת פונקציות ומחלקות ותפקידיהם 16](#_Toc101641785)

[6 סיכום אישי / רפלקציה 22](#_Toc101641786)

[7 מקורות מידע / ביבליוגרפיה 23](#_Toc101641787)

# ניהול שינויים

|  |  |  |  |
| --- | --- | --- | --- |
| **פעילות** | **גרסה** | **תכולה / שינוי** | **תאריך סיום** |
| יזום | 0.1 | הצעה ראשונית, ומילוי המבוא | 27.12.21 |
| אפיון | 0.2 | אפיון ארכיטקטורת הפרוייקט וסביבת העבודה | 31.12.21 |
| שלד | 1 | שלד פרוייקט עם פונקציונליות מינימלית | 7.2.22 |
| הוספת פונקציונליות | 1.1 | פרוייקט עם חיבור למסד נתונים עם ספרים משתמשים עם הגנה מינימלית. | 8.3.22 |
| עוד פונקציונליות | 1.2 | הוספת צ'אט הגנה והצגת ספרים בצורה לא מוגנת | 12.4.22 |
| מוצר ראשוני | 1.3 | הוספת התוכנה להצגת הספרים בצורה מאובטחת ואיבטוח כל מה שלא היה מאובטח | 23.4.22 |

**תכולה**

# מבוא

פרוייקט זה מממש דף אינטרנט אשר מנהל מערכת סיפרייה, כל אדם יכול להכנס לדף האינטרנט, להרשם, להסתכל על הספרים השונים, ולהשאיל אותם במידת הצורך. דף האינטרנט הוא בעצם דף אינטרנט של ספרייה כלשהי. קוד הפרוייקט של הסרבר פותר בעיית איחסון מידע גדול והחזרת המידע, ובמקרה זה זה אומר החזרת פרטי משתמש, על סוגי ספרים, עותקים שונים של הספרים, ועוד.

## תיאור המוצר המוגמר

המוצר הגמור הינו שרת אשר מקבל בקשות שונות ומטפל בהם, המשתמש מתחבר דרך ה- browser וגולש באתר. ישנם מספר סוגי משתמשים, כאלו שיכולים להעלות ספרים ועותקים וכאלו

שלא. לכל משתמש יש נתונים שונים כמו מספר הספרים שהשאיל, כמה זמן הוא השאיל, וכו'. כל אחד מתחבר, עם שם משתמש וסיסמא. בנוסף, חלק מהאתר הוא צ'אט, ישנם מספר של קבוצות אשר משתמשים יכולים להכנס אליהם וברגע שמתחברים אליהם יש מערכת צ'אט שבה אתה יכול לתקשר עם משתמשים אחרים באותה קבוצה בreal-time.

## נושא המחקר בפרויקט

נושא המחקר בפרוייקט זה הוא ניהול נכון כמות רבה של מידע עם מסד נתונים, מציאת המידע, ההחלצה של המידע, והכנסת המידע. הנושא בעיקר נבחר בגלל הפרוייקט ולא הפרוייקט בגלל הנושא, בפרוייקט עצמו יש מספר גדול של משתמשים וספרים, ולכן צריך גם לנהל את כל המידע הזה במסד נתונים. יש בעצם הרבה מידע שצריך לדעת איך לנהל, וצריך לנהל אותו.

מערכות ניהול מידע כאלו בדרך כלל הן או תוכנות פרטיות של עסקים בלי ממשק משתמש לאדם מבחוץ, או אתרים כמו Amazon, YouTube, וכו'. זה נושא חשוב ואני יודע שאני בעצמי משתמש במערכות כאלו הרבה מאוד. כל אחת היא שונה מהאחרת, יש כאלו שמושמשות בשביל שיתוף קבצים כמו סרטונים או שירים, כמו YouTube, ויש כאלו שמושמות בשביל ניהול דברים פיזיים ומחירה שלהם. הפרוייקט שלי נופל תחת הקבוצה השנייה, אני מנהל דברים פיזיים דרך המחשב ואז דרך מערכת השאלה מחלק את הספרים.

משום שיש וריאציה כזואת גדולה צריך גם חידושים ברעיון, כל רעיון הוא בשביל שימוש שונה אבל זה עדיין חשוב שיהיה וריאציה, והחידוש הגדול ביותר שלי הוא הוספת chat שעובד בReal-Time. בדרך כלל הפורומים שבהם אפשר לדבר בתוכנות כאלו הם לא בReal-Time בשביל לראות הודעות חדשות צריך לעשות refresh לדף וההודעות אינם מסודרות, זה לא Chat אלא מקום שאתה אומר מה אתה חושב על המוצר. בנוסף במוצר משאילים את הספרים לזמן מוגבל כלומר צריך דרך לתת לצרכן לקרוא את הספר לזמן מוגבר ואז לקחת ממנו את האפשרות אחרי שהזמן הזה נגמר.

# סביבת העבודה בפרויקט

שפת התכנות המרכזית שבה השתמשתי היאjava אבל כמובן שזה לא מספיק משום שזה אתר על תקשורת HTTPS שנמצא בbrowser-, הצתרכתי להשתמש בעוד כמה דרכים בשביל להראות את המידע ולכן ישנם קבצי HTML בשביל הlayout הכללי CSS בשביל העיצוב וJS בשביל האינטראקטיביות, ובמיוחד בשביל הchat. בנוסף גם השתמשתי בפייתון בשביל ממשק המשתמש בכדי לצפות בספרים אשר נמצאים רק לזמן מוגבל ומאובטחים.

את קוד הjava אני הרצתי מעל Tomcat בשביל שירוץ באינטרנט, בנוסף סביבת העבודה של הפרוייקט הייתה IntelliJ Ultimate, זאת הגירסה שמשלמים עליה. קילבתי אותה בחינם משום שאני סטודנט וחשבתי שיתאים יותר לעבוד עליה בשביל הפרוייקט הזה.

כל הגרסאות שבחרתי בהן הן בערך מאותה סיבה, בחרתי בגרסא החדשה ביותר משום שאלו הגרסאות שלמדתי איתם ובגלל שאלו השפות עם הכי הרבה פיצ'רים, הגרסאות הן:

JDK 17.01, אני משתמש בג'אווה 17 בפרוייקט, אין סיבה מיוחד פשוט מה שאני רגיל אליו ומה שאני יודע הכי טוב.

python 3.9, זאת גרסת הפייתון שאני מרגיש הכי נוח איתה משום שזאת השפה שאיתה למדנו, ואני כבר יודע איך רוב הדברים עובדים כבר. בחרתי בפייתון בכללי מאותה סיבה שבחרתי את הגרסה, זאת שפת התכנות שאיתה למדתי.

tomcat 9.0.591, השתמשתי בגרסא הזאת משום שזאת הגרסא שהתאימה לפרוייקט ולגרסאת הג'אווה.

JavaScript ECMAScript 2018, בחרתי בגרסא זו גם מאותן סיבות, זו הגרסה שלמדתי עליה ויש בה הכי הרבה פיצ'רים. וג'אווה סקריפט זו השפה שתמיד השתמשתי בה בשביל לבנות אתרים.

5HTML וCSS3, מאותן סיבות.

בנוסף לאלו אני משתמש גם בספריות שונות בשביל הדברים שצריכים עוד טיפה לוגיקה שאני לא יכול לעשות לבד.

הסיפריות של ג'אווה -

java.servlet – השתמשתי בספרייה הזאת בשביל יצירת הקשר, שליחת ההודעות, וקיבול ההודעות. לכן היא ספרייה מרכזית שאני לא יכול לעבוד בלעדיה. היא גם מנהלת את הפעילות האי סינכרונית בין תיפול הבקשות.

java.io – אני משתמש בספרייה הזאת בעיקר בשביל טיפול בקבצים ששמרתי על המחשב, כמו קובץ הPDF, המפתח אליו, המפתח למפתח, והתמונה של הספר.

java.security – הספרייה הזו נחוצה עבור HASHING שאני עושה במיוחד בשביל הסיסמאות כשאני מכניס אותם לתוך הDATABASE.

derby – אני משתמש בספרייה הזאת בשביל הניהול של הdatabase.

java.sql – זו הספרייה הדיפולטית של JAVA בשביל להשתמש במסדי נתונים ומשום שאין לי צרכים מיוחדים אין לי בעיה להשתמש בה.

java.util– אני בעיקר משתמש בספרייה הזאת בשביל ArrayList שהוא מאוד נוח בשביל לנהל מערכים דינמיים. בנוסף אני משתמש בה בשביל עוד דברים כמו Random וBase64

pdfbox – זו ספרייה נוחה בשביל ניהול של קבצי PDF בג'אווה ולכן אני משתמש בה.

java.nio.file – אני משתמש בספרייה הזאת בעיקר בשביל לקרוא קבצים בתור String ולא ביטים.

javax.crypto – הספרייה הזאת מאוד שימושית בשביל הצפנה סינכרונית של AES ואני משתמש בה בשביל ההצפנה של קובץ הPDF.

commons.codec – הספרייה הזאת שימושית בשביל לעשות Encode וDecode לHex.

commons.io – אני משתמש בספריה הזאת בשביל ליצור ולקרוא מקבצים.

javax.websocket – הספרייה הזאת משומשת בשביל ליצור Persistent connection עם הלקוח ואני משתמש בזה בשביל הצ'אט.

הסיפריות של javascript –

JQuery – אני משתמש בזה בשביל התקשורת עם היותר לוגיקה בצד הלקוח.

הסיפריות של python –

Hashlib – אני משתמש בספרייה הזאת בשביל לעשות hash לסיסמא כאשר אני שולח אותה.

Urllib – אני משתמש בספרייה הזאת בשביל ליצור בקשות https.

pdf2image – אני משתמש בספרייה הזאת בשביל לקרוא מקובץ pdf.

Kivi md – אני משתמש בספרייה הזאת בשביל הממשק הגרפי.

Os – אני משתמש בספרייה הזאת בשביל לבצע פקודות של הos ולמחוק קבצים.

Cryptodome – אני משתמש בתיקייה הזאת בשביל ההצפנה הסיכנרונית בעזרת AES.

Base64 – אני משתתמש בספרייה בשביל לעשות קידוד של Base64.

## טכנולוגיות בשימוש בפרויקט

בפרוייקט זה יש מספר טכנולוגיות שונות שנמצאות בשימוש, כל אחת מסיבה שונה. הטכנולוגיות הן:

שרת מרובה משתמשים – בכל רגע יכולים להתחבר מספר משתמשים לאתר האינטרנט, אפשר להגיע למצב כזה גם בלי תכנות מקבילי אבל בשביל הקלה על ההשהיות בתשובות זה דבר חשוב שהפרוייקט כן ישתמש בתכנות מקבילי שבזכותו הוא יכול לטפל בכמה בקשות באותו הזמן בצורה המהירה ביותר שאפשר.

מסדי נתונים – בפרוייקט ישנם שני מסדי נתונים, אחד מהם הוא האחד שמנהל את כל הספרים והמשתמשים במערכת, והשני שומר את כל השיחות והקבוצות המתקיימות בצ'אטים. מסדי הנתונים הם החלק המרכזי בפרוייקט, מציאת ושמירת מידע באופן יעיל, ולכן זה הוא חלק בלתי נפרד מהפרוייקט.

אבטחת מידע – חשוב להבטיח את המידע העובר באינטרנט, ובמיוחד המידע הרגיש, ולכן הפרוייקט משתמש בפרוטוקול להבטחת המידע העובר. בלי זה כל אחד יכול לראות את מה שעובר באינטרנט, לקלוט את הסיסמאות, את הספרים שאנשים רוצים להשאיל, וכו'. משום שאנחנו לא רוצים לפגוע בזכות לפרטיות של האנשים המשתמשים באתר איבטוח המידע העובר זה דבר נחוץ.

## מדריך למשתמש

אין הרבה דברים שהמשתמש צריך בשביל להריץ את הפרוייקט, כל מה שצריך זה חיבור לאינטרנט. החלק הגדול של הפרוייקט נמצא בתוך ה-Browser והחלק השני נמצא ב-client שהמשתמש יכול להוריד מתוך האתר. בשביל להריץ את התוכנה שמראה את הספרים אפשר להשתמש בInterperter שנמצא כבר בתוך קובץ הzip שמעבירים או אפשר להשתמש במה שיש כבר על המחשב.

אם המשתמש רוצה להשתמש בפייתון שיש לו עם המחשב כבר הוא יצתרך להוריד כמה סיפריות, בשביל להוריד אותם הוא צריך רק לפתוח את ה-Command Prompt ולכתוב את הפקודות הבאות:

pip install pdfplumber

pip install pycryptodomex

pip install pdf2image

pip install <https://github.com/kivymd/KivyMD/archive/master.zip>

בשביל להכנס לאתר צריך רק לגלוש אליו, ומשם מופיע מסך Log In אפשר או להכנס או להרשם למערכת על ידי כפתור הSign Up שמעביר אותך לדף ההרשמה, בנוסף יש גם כפתור של Log in as guest. אחרי שנרשמים ונכנסים למשתמש יש כמה אפשרויות פתוחות, נתחיל מהאפשרויות של האורח.

האורח כאשר הוא נכנס יכול אך ורק ללחוץ על כפתור החיפוש למעלה מצד ימין, האפרות לפתוח את הצ'אט ואת מסך הפרופיל לא פתוחים לא. מצד שמאל לכפתור החיפוש יש אפשרות להוסיף את מילות החיפוש לספר, אם לא מכניסים כלום זה פשוט מראה את כל הספרים שנמצאים במערכת. כאשר מראים את כל הספרים שנמצא אפשר ללחוץ עליהם ולראות את הפרטים על הספר וכל העותקים שיש. בתור האורח הוא יכול רק לראות אם השאילו את העותקים או לא ואין לו את היכולת להשאיל אותם.

בתור משתמש רגיל יש את האפשרות גם לפתוח את הצ'אט ואת מסך הפרופיל, בפרופיל יש לך את הכפתור להוריד את התוכנה של הספרים ויש את הרשימה של הספרים שהשאלת עם אפשרות להחזיר את הספרים מוקדם. בצ'אט יש אפשרות להתכתב עם אנשים אחרים המחוברים לצ'אט ב-Real Time כך שזה מראה את כל ההודעות ברגע ששולחים אותם. בתוך התוכנה המשנית יש לכל משתמש את האפשרות לפתוח את הספר ולדפדף בו.

האדמין יכול לעשות את כל מה שהמשתמש יכול לעשות רק שיכול גם להוסיף ספרים במסך הפרופיל שלו.

# אפיון דרישות וארכיטקטורת המערכת

## דרישות ושימושי מערכת – Use Cases



Diagram

Description automatically generated

במערכת ישנם שלושה רמות שונות של משתמשים וכל אחד עם הרשאות שונות. ברמה הראשונה ישנו את ה-Guest, הוא לא נרשם כלומר אנחנו לא יכולים לרשום אותו בבסיס הנתונים, לכן ההרשאה היחידה שיש לו זה להסתכל על הספרים השונים שיש בספרייה בלי להשאיל אותם או לראות את תוכנם.

סוג המשתמש השני הוא ה-User הרגיל, משתמש מרמה כזאת יכול לעשות את מה שהאורח יכול לעשות, אך הוא גם יכול להשאיל ספר ולצפות בתוכנו של הספר בתנאי שאף אחד לא השאיל את הספר לפניו.

סוג המשתמש השלישי והאחרון הוא ה-Admin. ביכולתו בנוסף להוסיף ספרים למערכת ואין לו הגבלות על השאלת ספרים כל עוד אף אחד אחר לא השאיל את הספר, זוהי רמת משתמש בעלת כוח גדול ולכן חייב להגדיר את המשתמשים שהם Librarian בצורה ידנית בבסיס הנתונים.

כל משתמש רגיל יכול להשאיל רק שלושה ספרים באותו הזמן ויש לו שבוע עם כל ספר שהוא משאיל לפני שנלקחת לו ההרשאה לצפות בספר.

בנוסף ל-Admin ול-User יש אפשרות להכנס לצ'אט ולשוחח שם.

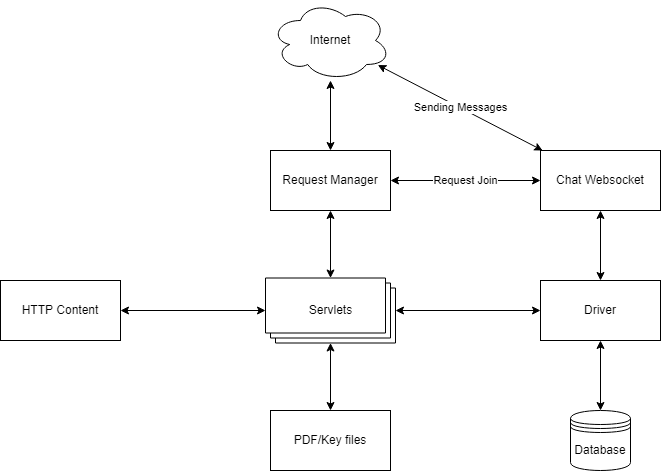
## סביבת הפרויקט – Eco – System

Graphical user interface, application

Description automatically generated

דיאגרמה זו מייצגת את הקשר בין השרת ללקוח והיכן איפה כל אחד מהם נמצא ביחס אחד לשני.

## ארכיטקטורת המערכת



הדיאגרמה הזאת מייצגת את החיבור של כל איבר בתוכנה אחד לשני, ברגע שנשלחת הודעה מהלקוח היא הולכת לRequest Manager ושם הוא מסדר את הבקשות לשני מודולים שונים האחד הוא הChat Websocket-, והשני הוא ה-Servlets, ה-Servlets מטפלים בבעיות מתוך פרוטוקול HTTP רגיל שהם non-persistent, בעוד שה-Chat Websocket מטפל בקשרים שנוצרים לאחר הבקשה להכנס. שני המודולים מנותקים בגלל שאחד מהם מטפל בקשרים persistent בעוד שהאחר מטפל בקשרי non-persistent (Servlets – non-persistent, Chat Websocket – persistent). שניהם נגשים אחר כך לDatabase- בשביל אימות נתונים ושימושים אחרים, וה-Servlets השונים ניגשים גם לקבצים שנמצאים בפרוייקט כמו הקבצים הסטטיים – קבצי הHTTP, והקבצים הלא סטטיים – קבצי המפתחות והPDF.

Request Manager – מקבל בקשות HTTP ומחזיר תשובות HTTP, הוא מפריד בין הבקשות ושולח אותם לServlets או לWebsocket המתאימים ומחזיר בחזרה את התשובות שהם נותנים לו.

Servlets – מקבל בקשות HTTP ומחזיר תשובות HTTP לפי הבקשה והמודולים השונים כמו הHTTP Conetent או מהDatabase.

Chat Websocket – מקבל קודם כל את בקשת ההתחברות הרגילה מהRequest Manager שהיא תחת פרותוקול HTTP עם header של upgrade מסויים ואז יוצר קשר משל עצמו עם כל אחד מהמשתמשים תחת קשר TCP/IP.

Driver – המודול הזה מקבל את כל הבקשות אל הDatabase מוצא להם תשובה ומחזיר את התשובה הזאת, אם הוא הצליח או לא או מה שהוא מצא. חלק מהתשובות הן תשובות של מספרים שמראים מה קרה בדיוק וחק מהם פשוט מחזיר את האובייקט שבוקש.

HTTP Content – זה כולל את כל הדברים שצריך בשביל שרת וובי כמו HTML/JSP או CSS או קבצי JS.

PDF/Key files – כולל את כל הקבצים של הספרים והדברים הקשורים אליהם כמו התמונות שלהם או את המפתחות בשביל לפתוח אותם.

## מסד הנתונים

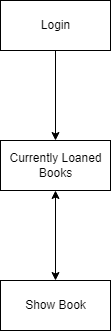
השתמשתי במסד נתונים SQL של Apache Derby משום שהוא מסד הנתונים שאני מכיר הכי טוב, בשבילו צריך להוריד את הספרייה derby.jar. יש לי מסד נתונים שיש בו 5 טבלאות, רובם מחוברים אחד לשני וכל אחד שומר משהו אחר לגבי הפרוייקט. בחרתי לשמור את הקבצים כמו הpdf בתוך הפרוייקט בתור קובץ במקום במסד הנתונים מטעמי יעילות.

הטבלאות שאני משתמש בהם הם –

1. USERS, הטבלא הזאת מכילה את כל המשתמשים בשרת.  
   בתוכה יש 4 תורים,  
   - EMAIL, האימייל של המשתמש, לא יכול להיות לשני משתמשים אותו אימייל.  
   - PASSWORD, הסיסמא של כל משתמש, לסיסמא עושים hash לפני שמכניסים אותה לתוך המסד נתונים  
   - USERNAME, שם המשתמש.  
   - AUTHORITY, מספר המייצג מה המשתמש בעצם, אורח משתמש רגיל או אדמין. אורח לא באמת נמצא במסד הנתונים אבל בבדיקה יש את האפשרות לזה שהיא 0
2. BOOKS, אני משתמש בטבלא הזאת בשביל לשמור את כל סוגי הספרים.  
   בתוכה יש 5 תורים,   
   - ISBN שזה בעצם קוד הספר, לכל ספר יש אחד שונה.  
   - BOOK-NAME שזה השם של הספר.  
   - AUTHOR כלומר שם הסופר.  
   - CATEGORY קטגרוית הספר.  
   - PUBLICATION\_YEAR שנת ההוצאה לאור של הספר מספרים.
3. COPIES, זאת טבלא שמתקשרת לBOOKS משום שהיא מכילה עותקים של ספרים מתוך הספרים שנמצאים בתוך הטבלא BOOKS.  
   בתוכה יש 3 תורים,  
   - COPY-CODE, כלומר הקוד של העותק, לכל עותק יש קוד שונה.  
   - ISBN, קוד הספר שהוא שייך אליו.  
   - LOANED, משתנה Boolean שקובע אם השאילו את העותק או לא.
4. LOANS, זאת טבלא שמקשרת בין הUSERS לCOPIES. היא קובעת מי השאיל איזה עותק ומתי. הטבלא הזאת יכלה להתחבר לCOPIES לטבלא אחת אבל אני מאמין בהפרד ומשול, ולכן אני חושב שמתאים שתהיה לטבלא הזאת טבלא במיוחד.  
   בתוכה 3 תורים,  
   - COPY-CODE, שמקשר אל העותק שנשאל, זה המשתנה הייחודי בטבלא.  
   - EMAIL, מקשר אל טבלאת המשתמשים.  
   - DATE\_LOANED, שאומר מתי כל עותק נשאל זה משתנה מספרי.
5. GROUPS, הטבלא הזאת מכילה את הקבוצות ואת המשתמשים המחוברים אליה.  
   בתוכה יש 7 תורים,  
   - CODE, המשתנה היחודי.  
   - GROUP\_NAME, השם של הקבוצה.  
   - ADMIN, האימייל של המשתמש שיש לו יכולות.  
   - USER2-5, זה המשתמשים שיכולים להכנס.  
     
   סך הכל יכולים להיות 5 משתמשים.

## ממשק משתמש - GUI

בחרתי בשילוב בין ממשק וובי וממשק גרפי פשוט לתוכנה להצגת הקבצים. הממשק הוובי משומש בשביל הכל חוץ מהצגת הקבצים שצריך היה לעשות בנפרד בשביל לאבטח אותם. בצד השרת אין שום דבר מיוחד, רק להריץ והכל עובד לבד.



זוהי דיאגרמת הזרימה של תוכנת ההצגה, אין הרבה מה לעשות שם משום שהיא רק נמצאת בשביל להציג את הספרים בלי פונקציונליות מעבר.

Diagram

Description automatically generated

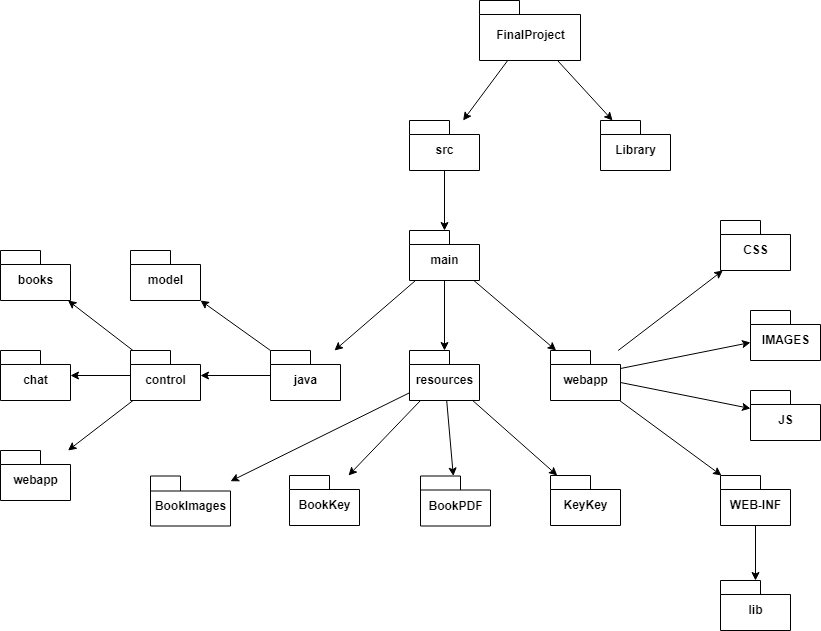
הדיאגרמה הזאת לא מדויקת, משום שבכל מודול חוץ ממסך הכניסה וההרשמות יש באר למעלה שיש בו לינק למודולים האחרים, הלינקים הם לProfile Page, Chat, Main Page ולBook Search. מהBook Search אפשר להגיע לדף של כל ספר ומשם אם אתה עומד בתנאים אתה יכול להשאיל את העותקים של הספרים. כל עותק יכול להשאל על ידי אדם אחד כל פעם.

# מדריך למפתח

## הרצת השרת

בשביל הרצת השרת צריך לוודאות שכמה דברים נכונים, קודם כל בקישור לDatabase צריך לוודאות שבקוד אנחנו עושים reference למקום הנכון שמסד הנתונים נמצא בו, המיקום הזה יהיה "jdbc:derby:[PATH TO PROJECT]/Library". השורה שצריך לשנות נמצאת בDBUtil.java בשורה 19. חוץ מזה צריך לוודאות שגירסאת הTomcat היא 9.0.591 וגרסאת הג'אווה היא 17. בנוסף בשביל שHttps יעבוד לא צריך לשנות כלום בפרוייקט עצמו משום שזה בהגדרות של tomcat. בסוף הפרוייקט בביליוגרפיה הוספתי לינק לאתר שמראה איך לעשות את זה.

## דיאגרמת התיקיות בפרוייקט ותפקידיהם



בפרוייקט יש כמות טובה של תיקיות, יש את התיקייה הראשית שהיא של הפרוייקט שקוראים לה Final Project, היא מאכסנת את כל הפרוייקט.

בתוכה נמצאות התיקיות Library וsrc ועוד כמה תיקיות שנוצרות באופן אוטומטי. בתוך Library יש את בסיס הנתונים, בתוכה יש עוד כמה תיקיות שנוצרו באופן אוטומטי כאשר יוצרים את בסיס הנתונים. בתוך התיקיה src יש את התיקיה main ואת התיקיה test, התיקיה test נוצרת באופן אוטומטי ולא נגעתי בה. בתוך התיקיה main יש כל הקבצים ואת התוכן של הפרוייקט. בתוך main יש שלושה תיקיות.

* java, שבתוכה נמצא כל הקוד של ג'אווה.
* resources, עם כל הקבצים הלא סטטיים של הפרוייקט.
* וwebapp עם הקבצים הסטטיים של ממשק המשתמש.

בתוך java יש עוד שתי תיקיות, model וcontrol.

בתוך model נמצאים כל המחלקות בשביל אובייקטים שמשומשות בפרוייקט כמו User או Book, אבל נמצאות שם גם מחלקות אבסטרקטיות שמשומשות רק בשביל הפונקציות שבהן, כמו AESCrypt או DBUtil.

בתוך control נמצאות עוד שלושה תיקיות –

* books – שמנהלת את כל הServlets שעונים לתוכנה של הPDF.
* chat – שמנהל את הWebsocket לchat.
* webapp – שמנהל את כל הServlets שעונים לבקשות של הממשק הוובי.

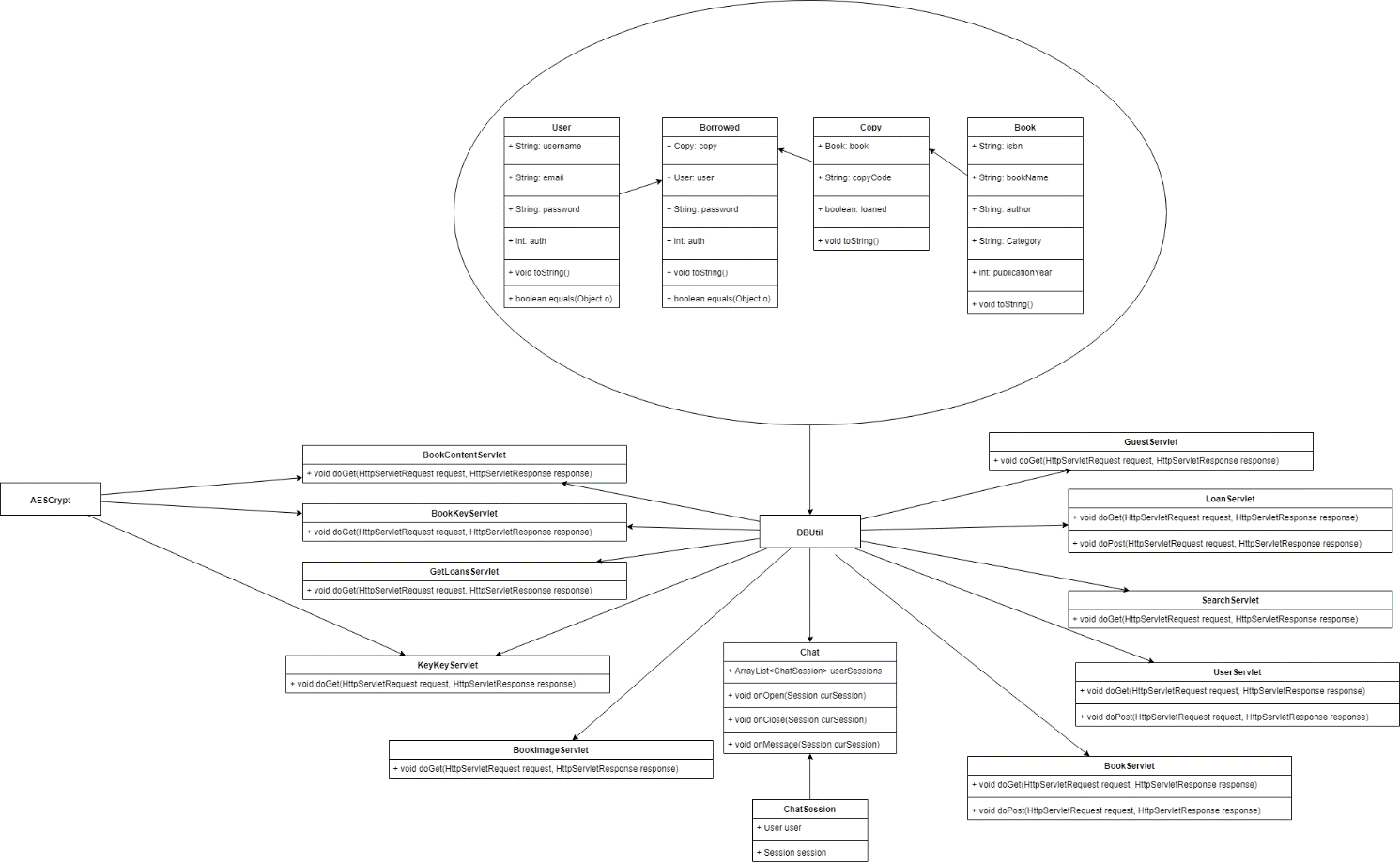
בתוך resources ישנם ארבעה תיקיות –

* BookImages – יש את הכרכים של הספרים.
* BookKey – המפתחות של הספרים, זו תקייה של קבצים זמניים שרק הייתי צריך בשביל שאוכל לשלוח גם את הספר וגם את המפתח באותו הזמן.
* BookPDF – יש בו את קבצי הPDF של הספרים.
* KeyKey – הקבצים של המפתחות למתפחות של הספרים.

בwebapp יש גם תיקיות וגם קבצים, יש את כל קבצי הJSP של הממשק הוובי ועוד ארבעה תיקיות –

* CSS – כל קבצי הCSS שקבצי הJSP משתמשים בהם.
* IMAGES – התמונות שיש בממשק, כמו תמונת הרקע.
* JS – קבצי הJavaScript שקבצי הJSP משתמשים בהם.
* WEB-INF – יש כאן קובץ web.xml שכולל את המבנה את והתכונות של האתר, לא הוספתי בו יותר ממה שיש אוטומטית. ובנוסף יש כאן תיקיית lib שנמצא בה כל הסיפריות החיצוניות שמשומשות בפרוייקט.

## דיאגרמת UML של כל מחלקות הפרויקט והתלויות ביניהן



## רשימת פונקציות ומחלקות ותפקידיהם

**רשימת הקבצים -**

|  |  |  |
| --- | --- | --- |
| Location | Use | File Name |
| src/main/java/control/books | Returns encrypted PDF of the book. | BookContentServlet |
| src/main/java/control/books | Returns encrypted key to the PDF file. | BookKeyServlet |
| src/main/java/control/books | Returns names and ISBN of all loaned books by the user. | GetLoansServlet |
| src/main/java/control/books | Returns the key to the encrypted key for the pdf. | KeyKeyServlet |
| src/main/java/control/chat | Handles all Chat functions. | Chat |
| src/main/java/control/webapp | Returns the image of a book. | BookImageServlet |
| src/main/java/control/webapp | Handles requests. | BookServlet |
| src/main/java/control/webapp | Logs the user in as a Guest. | GuestServlet |
| src/main/java/control/webapp | Handles requests. | LoanServlet |
| src/main/java/control/webapp | Forwards to search.jsp with the search parameters. | SearchServlet |
| src/main/java/control/webapp | Handles requests. | UserServlet |
| src/main/java/control/model | Handles encryptions and decryptions. | AESCrypt |
| src/main/java/control/model | Class of book object. | Book |
| src/main/java/control/model | Class of Borrowed object. | Borrowed |
| src/main/java/control/model | Class of ChatSession object. | ChatSession |
| src/main/java/control/model | Class of Copy object. | Copy |
| src/main/java/control/model | Class of Group object. | Group |
| src/main/java/control/model | Class of User object. | User |
| src/main/java/control/model | Handles utility functions for database. | DBUtil |
| src/main/webapp/CSS | CSS file for chat jsp. | chat.css |
| src/main/webapp/CSS | CSS file for pop ups. | modal.css |
| src/main/webapp/CSS | General CSS file. | style.css |
| src/main/webapp/IMAGES | Background image. | background.jpg |
| src/main/webapp/IMAGES | Icon image. | icon.png |
| src/main/webapp/IMAGES | The image in the login and signup pages. | login-icon.png |
| src/main/webapp | The web page containing information about the book. | book.jsp |
| src/main/webapp | The web page containing the chat. | chat.jsp |
| src/main/webapp | The login webpage. | index.jsp |
| src/main/webapp | The profile page. | profile.jsp |
| src/main/webapp | The register web page. | register.jsp |
| src/main/webapp | The webpage for search. | search.jsp |
| src/main/webapp | The home page. | welcome.jsp |
| src/main/webapp/WEB-INF | The commons codec library. | commons-codec-1.15.jar |
| src/main/webapp/WEB-INF | The commons fileupload library. | commons-fileupload-1.4.jar |
| src/main/webapp/WEB-INF | The commons io library. | commons-io-2.11.0.jar |
| src/main/webapp/WEB-INF | The derby library. | derby.jar |
| src/main/webapp/WEB-INF | The pdfbox library. | pdfbox-app-3.0.0-alpha2.jar |

**רשימת הפונקציות -**

|  |  |  |
| --- | --- | --- |
| Under what class | What it does | Function Signature |
| Chat.java | Gets the session of a user that would like to connect, checks if the user is valid, and adds him to the list of current connections. | void onOpen(Session curSession) |
| Chat.java | Removes user from the list of current connections. | void onClose(Session curSession) |
| Chat.java | Sends all users in the room the message. | void onMessage(String message, Session curSession) |
| BookContentServlet.java | Gets the ISBN of the book, and the email and password of the user. Returns an encrypted version of the book. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| BookKeyServlet.java | Gets the ISBN of the book, and the email and password of the user. Returns an encrypted version of the key for the book. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| GetLoansServlet.java | Gets the email of the user and returns the ISBN and names of the books the user has loaned | void doGet(HttpServletRequest request, HttpsServletResponse) |
| KeyKeyServlet.java | Gets the ISBN of the book, and the email and password of the user. Returns the key for the key. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| BookImageServlet.java | Gets the isbn of the book and returns the cover of the book. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| BookServlet.java | Gets the isbn of the book and forwards to the book.jsp page. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| BookServlet.java | Gets the ISBN, name, author, category, year, number of copies, pdf, cover of the book and uploads them to the database and files under the name of the ISBN. | void doPost(HttpServletRequest request, HttpsServletResponse) |
| GuestServlet.java | Forwards the user as a guest. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| LoanServlet.java | Gets a copycode and the user and loans the book if it isn’t already loaned and the user meets the conditions to loan a book. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| LoanServlet.java | Gets a copycode and the user and removes the loan from the user if it is loaned by him. | void doPost(HttpServletRequest request, HttpsServletResponse) |
| SearchServlet.java | Gets a description of the book and forwards it to search.jsp which find the books that match and shows them. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| UserServlet.java | Gets email and password and logs in the user if it matches the data in the database. It logs them in by adding the User object as session and forwarding to the homepage. | void doGet(HttpServletRequest request, HttpsServletResponse) |
| UserServlet.java | Get email, username and password and adds the user to the database if there isn’t a user with the email already used. | void doPost(HttpServletRequest request, HttpsServletResponse) |
| AESCrypt.java | Gets a key and a byte array and ciphers them using the AES function. It returns the cipher | byte[] encrypt(String key, byte[] message) |
| AESCrypt.java | Returns a random 128-bit AES key | String GenKey() |
| AESCrypt.java | Get a key and a file and saves the key as a file. | String SaveKey() |
| DBUtil.java | Returns a connection to the database, if there already is one it returns it, if there isn’t it make a new one and saves it. | Connection GetConnection() |
| DBUtil.java | Gets a user and returns whether it already is saved in the database. | boolean UserExists(User u) |
| DBUtil.java | Gets an email and returns the User that matches that email. | User GetUser(String email) |
| DBUtil.java | Gets username email and password and adds them to the database. Returns 0 if it failed but didn’t throw an exception for one reason or another, 1 if there is one, and 2 if it successfully added them to the database. | int AddUser(String username, String email, String password) |
| DBUtil.java | Gets an email of a user and returns all the groups he is a part of | ArrayList<Group> GetGroup(String email) |
| DBUtil.java | Gets a code for a group and an email and returns 0 if it failed there is no such group, 1 if there is but it’s full, and 2 if the user was successfully added. | int JoinGroup(String code, String email) |
| DBUtil.java | Gets an email and returns an ArrayList of all copies the user has loaned. | ArrayList<Borrowed> GetLoan(String email) |
| DBUtil.java | Gets a copycode and returns the borrowed object corresponding to it. | Borrowed GetLoanCopyCode(String copyCode) |
| DBUtil.java | Gets a copycode and a user and removes the loan, returns whether it was successfully removed or not. | boolean RemoveLoan(String copyCode, User u) |
| DBUtil.java | Gets a copycode and returns the copy of the book it corresponds to. | Copy GetCopy(String copyCode) |
| DBUtil.java | Gets an ISBN and returns all copies corresponding to that ISBN | ArrayList<Copy> GetCopyISBN(String isbn) |
| DBUtil.java | Gets an ISBN, name, name of author, category, publication year, and the number of copies and adds those copies to the database. returns 1 if there already is a book and it just added copies, and 2 if it also added the book. | int AddCopies(String isbn, String name, String author, String category, int year, int numOfCopies) |
| DBUtil.java | Gets a Book object and the number of copies and adds that number of copies of the book to an ArrayList and returns it. | ArrayList<Copy> MakeCopies(Book book, int numOfCopies) |
| DBUtil.java | Returns the largest copyCode of all book copies | int MaxCopyCode() |
| DBUtil.java | Gets ISBN of a book and returns a book object corresponding to that isbn. | Book GetBook(String isbn) |
|  | Gets a description of the book and returns all books matching that description. | ArrayList<Book> FindBooks(String desc) |
| DBUtil.java | Gets a copyCode and a User object and loans the book, returns 0 if the user doesn’t exist, 1 if the copy is already loaned and 2 if it was successful. | int LoanBook(String copyCode, String email) |
| DBUtil.java | Gets and String and hashes is using SHA-256. | String Hash(String str) |
| index.jsp  register.jsp | Gets a string and returns the sha256 hash of it | sha256(message) |

# סיכום אישי / רפלקציה

העבודה בפרוייקט הזה הייתה משהו חדש בשבילי, זה היה הרבה פחות לכתוב קוד והרבה יותר לחפש באינטרנט ולהבין קוד של אנשים אחרים ולהתאים אותו לקוד שלי. זה לא משהו שאני רגיל אליו מהשיעורים שלנו בבית ספר משום שבביתספר בדרך כלל הבעיות היו בעיות של מחשבה ופחות ידע בנושא משום שלמדנו הכל בכיתה כבר. היו לי גם הרבה יותר בעיות לא ממש ברורות משום שלא כל נושא וספרייה שכתבתי לפיהם היו פורפולרים ולכן היו הרבה בעיות שהיו קשה מאוד לפתור.

אני מאמין שלמדתי איך לכתוב קוד וללמוד מהאינטרנט הרבה יותר טוב, יש הבדל ענק בין איך שכותבים את השאלות ואיזה תשובות מוצאים ולכן צריך לדעת לא רק איך להבין את המידע אלא גם איך לחפש את המידע.

במהלך הפיתוח של הפרוייקט היו לי מספר בעיות שהיה לי בעיה עם לפתור, כמו להראות את הספר. מלחתחילה ניסיתי לפתור את זה בתוך הbrowser והיו לי בעיות עם זה, אני לבסוף התייאצתי עם חבר של אח שלי והוא הציע לי לעשות את זה בתוך תוכנה בנפרד שאני אשלח דרך האתר ושם אראה את תוכן הספרים. בערך בכל שלב ושלב הייתה לי בעיה כזאת או אחרת שהיה קשה להבין ולכן כל פעם שהצלחתי זה הרגיש מצויין. עוד בעיה לדוגמא הייתי לסנכרן את הencrypt וdecrypt בסרבר ובלקוח, לבסוף חיפשתי באינטרנט בשביל שני פונקציות שעובדות גם על ג'אווה וגם על פייתון וזה פתר את הבעיה, אחרי כמה שינויים שם כדי שיתאימו לצרכים שלי הכל עבד כמו שרציתי בלי יותר מדי בעיות.

אילו הייתי מתחיל את הפרוייקט היום לא הייתי עושה יותר מדי שונה אבל החלט הכל היה עובד הרבה יותר חלק, משום שאני יודע את מה שאני יודע עכשיו הייתי מספיק הרבה יותר והייתי יכול ליצור מוצר גמור יותר ואני מאמין שאם הייתי מנסה עוד פעם הייתי מצליח לשלוח את הספר דרך האתר בצורה מאובטחת משום שאני חושב שהבנתי מה הבעיה הייתה.

מלחתחילה בפרוייקט עשיתי הרבה שינויים. התחלתי מפרוייקט על השמעת שירים כמו Spotify והחלטתי לשנות את הפרוייקט לנושא שהוא עכשיו אבל גם אחרי זה החלטתי לעשות את זה בפייתון, לאחר כמה זמן אני למדתי יותר על ג'אווה והחלטתי שזה יהיה הרבה יותר מתאים לפרוייקט משום שזה הרבה יותר בנוי ויותר מסודר. במיוחד שעם הדברים שהשתמשתי בהם הרבה מהדברים שעשיתי הם נעשים אוטומטית, כמו מבנה הפרוייקט, ניהול הHTTPS, ניהול הבקשות ולאן כל אחד מהם הולך וכו'.

# מקורות מידע / ביבליוגרפיה

כל השאלות והתשובות שלי ושל אנשים אחרים נענו כאן –

<https://stackoverflow.com/>

מכאן הבאתי את הפונקציות המקוריות של הצפנת הAES –

<https://github.com/ijl20/python_java_crypto>

מכאן מצאתי איך להגדיר את tomcat נכון בשביל קשר https –

<https://tomcat.apache.org/tomcat-9.0-doc/ssl-howto.html>

מכאן למדתי עוד קצת על pycryptodome ובנוסף מצאתי פתרון לבעיה שהייתה –

<https://pycryptodome.readthedocs.io/en/latest/src/installation.html>

שאר מקורות המידע שלקחתי מהם היו מפה לפה, כלומר משיעור בבית ספר או דברים שלמדתי לפני כן. בנוסף גם התעזרתי עם שאלות באנשים אחרים.